Starter Kit

Welcome to Project Launchpad , the official UI Starter Kit for the Play+ Design System .

This isn’t just a boilerplate. It’s an opinionated, production-grade foundation designed to make your applications secure, fast, accessible, and maintainable from day one. Rooted in our philosophy of Design that Breathes , Launchpad empowers your team to focus on crafting emotionally resonant, adaptive user experiences—while we handle the rest.

Two Paths to Launch

Golden Path — For New Projects

This is the recommended path for all new applications. It provides a fully integrated, production-ready foundation in minutes.

Angular Setup

Uplift Path — For Existing Projects

Use this path to incrementally adopt Play+ best practices in legacy or ongoing projects. Install only the helpers you need.

Package	Helper	Description	Install Command
@playplus/tokens	—	Design Tokens for spacing, typography, theming	npm install @playplus/tokens
@playplus/security	playguard	Secure API proxy, input sanitization, auth flows	npm install @playplus/security
@playplus/logging	playlog	Structured, environment-aware logging	npm install @playplus/logging

Package	Helper	Description	Install Command
@playplus/storage	playcache	Safe localStorage API with TTL & serialization	npm install @playplus/storage
@playplus/features	playfeature	Feature flags, A/B testing, canary releases	npm install @playplus/features
@playplus/eslint-config	playlint	Linting rules aligned with Play+	npm install --save-dev @playplus/eslint-config
@playplus/testing-config	playtest	Testing setup for unit + CI	npm install --save-dev @playplus/testing-config

The Play+ Helper System

Each helper addresses a critical non-functional requirement and works seamlessly with others when using the Golden Path.

Helper	Purpose
playguard	Security : Prevents XSS, enforces secure API access, sanitizes input
playlog	Logging : Structured logs, secret redaction, remote transport support
playperf	Performance : Performance budgets (LCP/CLS), lazy loading utilities
playa11y	Accessibility : WCAG 2.2 AA compliance, focus helpers, screen reader support
playerror	Error Handling : Global error boundaries, graceful UI fallback, telemetry
playcache	Storage : Safer localStorage API, with automatic TTL & format validation

Helper	Purpose
playfeature	Feature Flags : Controlled rollouts, toggles, user targeting

What Launchpad Covers

The starter kit comes pre-configured with best practices across these critical domains:

- Folder Organization
- API Handling
- Logging Practices
- Security Best Practices
- Linting Rules
- Unit Testing Strategy
- Local Storage Guidelines
- Performance
- Accessibility
- Feature Flags
- Error Handling
- State Management
- Environment Configuration

Each of these is backed by detailed guides within the Play+ documentation.

Why We Went the Extra Mile

Why These Helpers Exist

Modern web apps are complex. Developers shouldn’t have to reinvent solutions for cross-cutting concerns like auth, logging, error recovery, accessibility, or performance every time they start a new project. That’s why we created Play+ Helpers .

Each helper solves a problem that every production-grade app eventually faces—but solves it once, in a secure, scalable, and tested way.

With the Golden Path, you don’t need to:

- Manually configure logging, performance budgets, or WCAG audits

- Build custom retry logic or feature flag infra
- Worry about storing sensitive data incorrectly in localStorage
- Maintain scattered error reporting or auth wrappers

Everything is pre-integrated and tested—so your team can stay in flow.

What You Get for Free

- A Fully Integrated System Every helper is wired into the others. Example: → API calls via apiProxy are secured with playguard → Failures are caught by playerror → Logged contextually by playlog → UI feedback provided using playperf degradation patterns This works automatically, with no extra setup.
- Zero-Config Best Practices Preconfigured: ESLint & Prettier Testing tools Security headers Performance budgets
- Developer Velocity Developers spend more time building features, less time on boilerplate, configuration, or fixing common pitfalls.
- Guaranteed Consistency Teams share conventions and architectural patterns from the start, simplifying onboarding and long-term maintenance.
- Seamless Upgrades New helpers and improvements can be adopted with minimal effort thanks to standardized hooks.

The Bottom Line

- Use the Uplift Path to modernize legacy codebases.
- For anything new, choose the Golden Path .